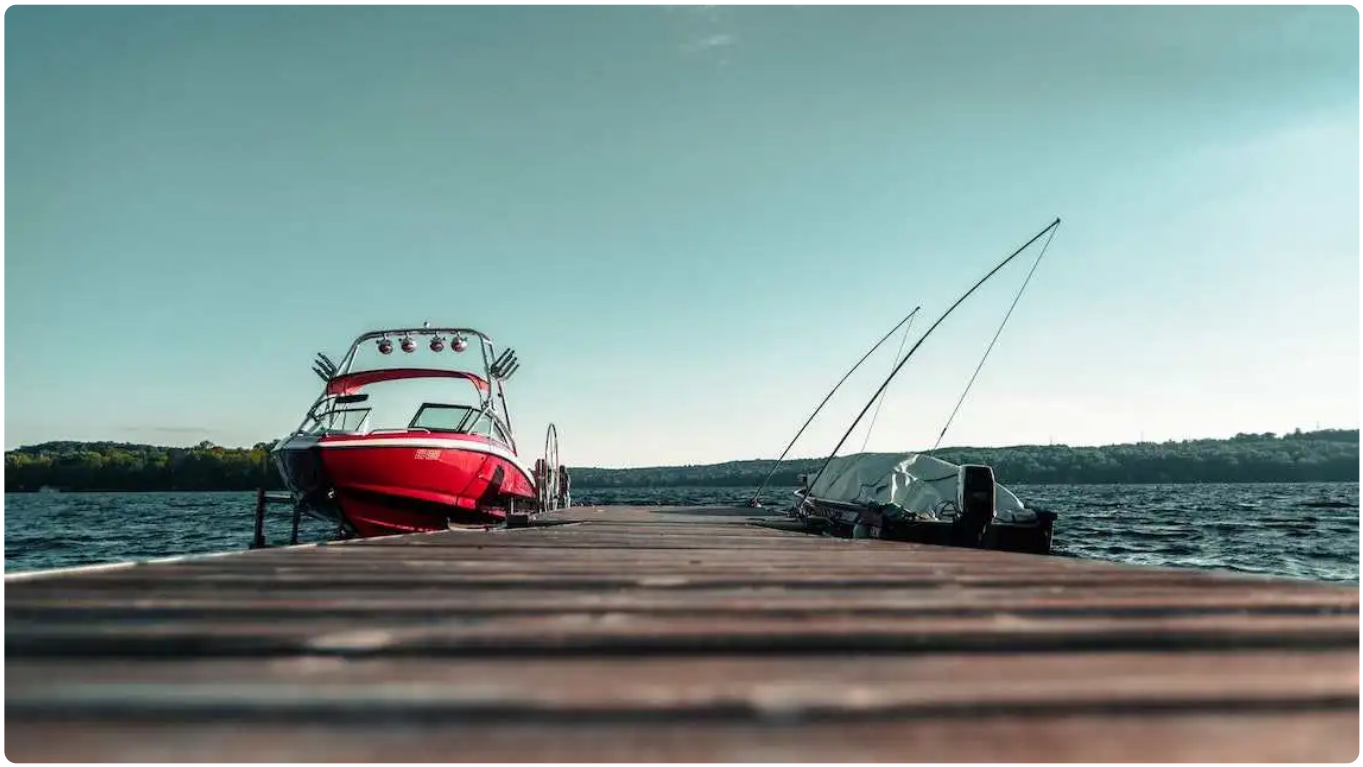
01｜ROI价值内核：自动化测试的价值可以量化么？

2022-03-21 柳胜

《自动化测试高手课》 课程介绍 >



讲述：柳胜

时长 21:21 大小 19.56M

你好，我是柳胜。

作为测试人员，我们都想做好自动化测试，但是每个行业都有自己的规律，也就是说常说的道，自动化测试也有自己的道。所以，在这个模块，我们的目标是了解自动化测试的道是什么，怎么能运用它让自己的测试工作更加有成效。

今天是价值篇的第一讲，我们先来弄清楚自动化测试的价值究竟是什么？看到这你可能有点困惑，自动化测试有那么多公司都在搞，自然是有价值的啊，有啥可讨论的呢？

其实这个问题非常关键，在开始工作之前，要把我们的工作价值想清楚，后续工作才能事半功倍。我列几个工作中我们频繁听到的问题，你会更有感触。

（上级沟通）“产品要上线了，QA 人手紧，能不能搞一下测试自动化，减少点人手？”

（调动人手）“什么？你还要再增加 2 个自动化测试开发工程师来完成这个项目，他们都要做什么？”

（工作述职）“听说你开发了个什么自动化脚本，它给公司带来了什么价值？用量化的数据给我讲一讲！”

这样的问句是不是似曾相识？其实它们都指向了一个硬核问题“自动化测试项目的价值是什么？”

在这节课，我要和你捋一下，为什么要做自动化测试，并且带你找到度量它价值的方法。掌握了这些，就能对自己的工作目标更清楚、更有信心，别人问到的时候，我们也能讲清楚、说明白，得到了团队理解工作将事半功倍。

为什么要搞自动化测试

开篇词中我提了出海捕鱼的场景，不只自动化测试，整个测试工作就像织网一样，会有弹性的空间，网眼大了、小了，捕到多少鱼，这些都有不确定性，但是这个不确定性又关系到成本和收益这些敏感问题，这是测试工作的一个特点。

我曾经跟我的团队说过 “咱们做测试工作，甭管用什么方法和技术，目标就是用最小的成本，得出对软件质量最大的确定性结论”。

自动化测试也面临相同的困难，为了解决这样的不确定，我们有必要好好分析一下，自动化的成本和收益究竟怎么算？

如果感觉这样问还有些抽象，我们不妨换个问法，自动化测试实施之前和之后，自动化带来的改变是什么？为了进一步完善思路，我们结合一个更具体的例子来做推理、估算。

这个例子是：一个 Web UI 订火车票的软件，成功订一张火车票这个测试案例，要做自动化所花费的成本、还有得到的收益，会是多少呢？

自动化实施之前，测试案例的执行要靠手工完成，一个工程师需要花费 0.5 个小时，运行完登录、订车票，查看数据库这样一个测试流程。

而自动化测试实施之后，流程可以用 Selenium 脚本自动完成，原先手工测试半个小时的工作量就省下来了。那么，省下来的这半个小时这就是自动化测试带来的价值。对不对？

对，但还不全面，我们衡量效益，不应该只看回报，还要看成本，我们要算上开发自动化测试花费的成本。为了开发 Selenium 订火车票的这个脚本，自动化测试工程师花费了 1 天的时间，合计就是 8 个小时。

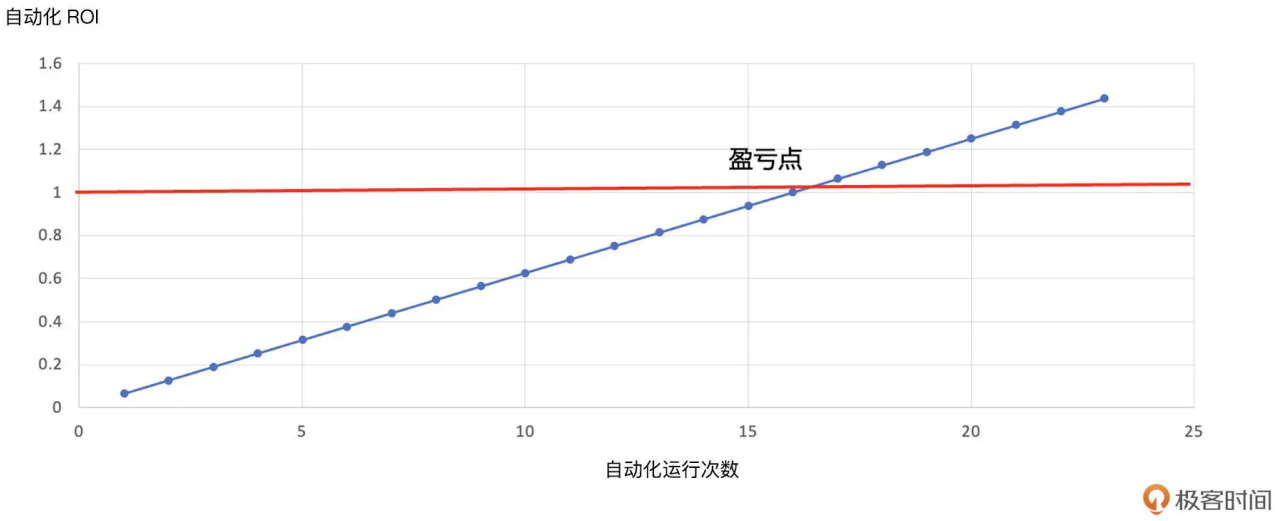
现在，这个 Selenium 自动化测试案例，它的投资收益比 ROI 应该这么计算（产出和投入都用时间作为单位）：

产出 / 投入 = 0.5/8= 0.0625

结果不到 7%。哇，投入了 8 个小时，才收获了 0.5 个小时。如果这是一项投资的话，那肯定是亏本的。哪个公司愿意做这样的买卖呢？

但是，上面的公式只计算了运行一次自动化测试案例的 ROI。实际上，自动化测试案例开发出来后，肯定不止运行一次的。多运行一次，就会多节省下来一份工作量，如果用 n 来指代运行次数，t 指代单次测试时间，现在的产出变成了 n*t，n 越大，产出就会越大。

那上面订火车票案例，运行多少次才能收回成本呢？1/0.0625=16，只要这个 selenium 脚本运行超过 16 次，我们就可以让 ROI=1，收支平衡，收回成本了。



运行次数&ROI关系示意图

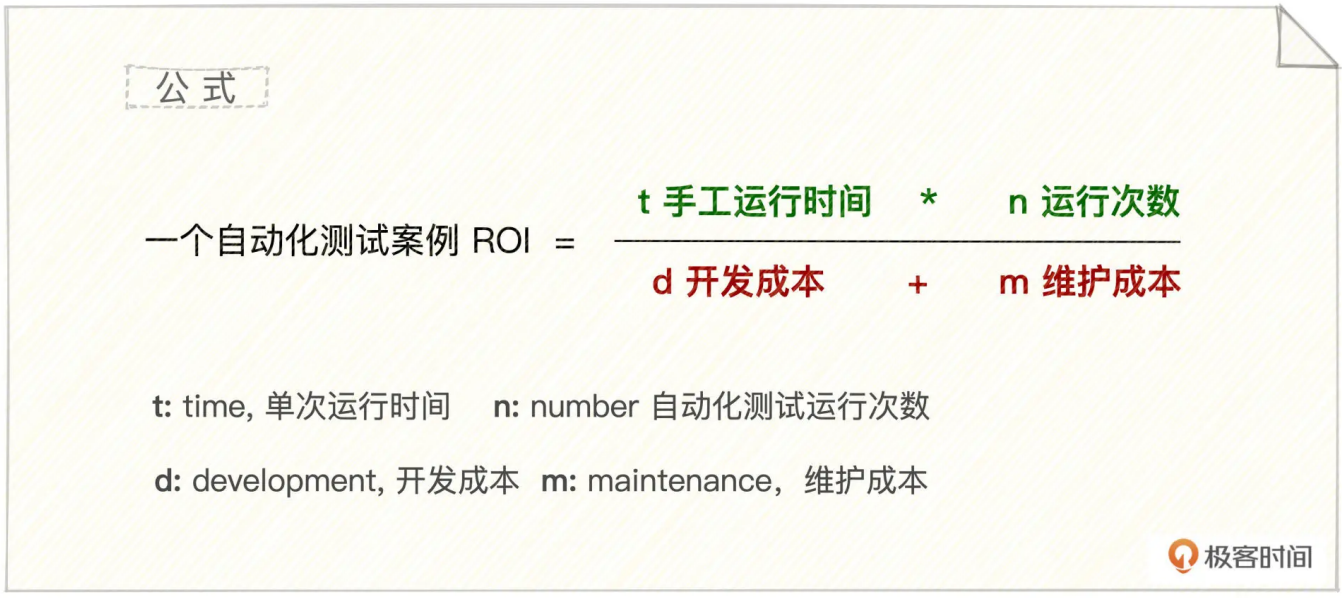
太棒了，现在你就可以对公司说：“我的自动化测试收益是可以量化的，只要我保证开发出来的脚本，能运行超过 16 次，就是为公司省钱了。”

且慢，还有一笔账没有算，除了开发成本，还有维护成本。自动化测试开发出来后，还需要维护版本升级、诊断错误、优化结构等等的工作，这笔成本是需要持续投入的。

现在这个 Selenium 在它线上运营生命周期内的 ROI 计算公式，变成了后面这样：

产出 / 投入 = 0.5*N/(8+ 维护成本）

我们把它提炼成一个计算公式，就是：



ROI计算公式

这个公式很简单，但仔细揣摩可以推导出几个有意思的结论。

1.ROI 大于 1 就是赚了，小于 1 就是亏了。那么，给定一个测试案例，要不要对它做自动化，判断的依据是（自动化测试）预期 ROI 至少要大于 1。

2. 自动化测试是一个长收益模式。在理想情况下，是一次性投入（投入为开发成本），之后每运行一次，就会增加一份产出。所以，时间越长，次数越多，收到的回报就会越大。

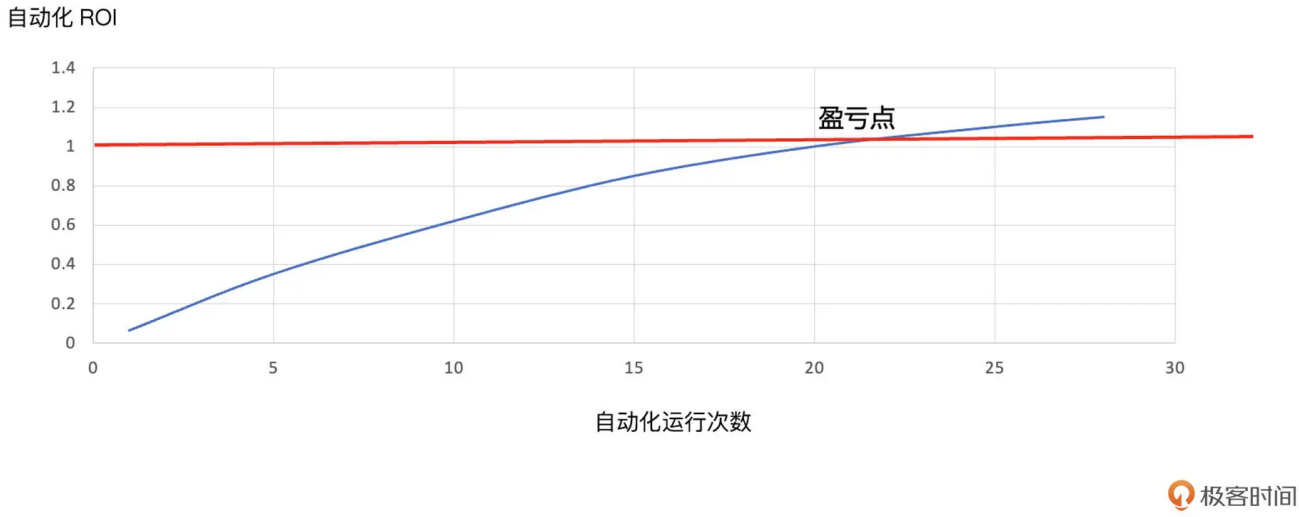
3. 关于开发成本（包括开发成本 d 和维护成本 m），类似估算软件开发工作量，代码行法、功能点法，我们也可以引入到估算开发工作量里，比较好掌握。**但维护成本就有点模**

糊了，这里包含了多种可变因素，是自动化测试项目风险的主要来源。

维护成本来自于多个地方。一段代码从产生以后，就会持续产生维护工作量，而且，因为存在架构腐化等问题，维护工作量增加速度是以非线性来增长的。

到了最后，一个陈旧的老破系统，加入一个新功能需要写 10 行代码，只要花 5 分钟。但是搞清楚这 10 行代码，应该加到哪个文件里，要花费 3 天时间。在这种时候，这个软件系统就已经不可维护了，它要寿终正寝了。自动化测试代码的维护成本更复杂，不仅面临着腐化的问题，还有被测产品更新带来的维护等等。

所以，在实践中，你看不到前面图里，那样简单漂亮的 ROI 直线，它会表现为一段曲线：自动化的 ROI 增长速度，要比运行次数增长慢一些，直到最后，每运行一次，花费的维护工作量，比节省的工作量还多，自动化就该退休了，也就是下线，重构完了再上线。



运行次数&ROI关系示意图2.0版

这里我想提醒你注意，**ROI 模型提供的是一种自动化测试投资收益比的量化思路，方便我们明确哪些因素影响着自动化测试效益。**不可能存在一个万能的公式，把参数往里一带，就会算出 ROI 的数字如果世间的事都这么简单，还需人类干什么。我们需要做的是尽可能量化，你对量化的了解越多，对自动化测试的理解就会更加深入全面。

做不做自动化测试，能用数据说话么？

从 ROI 的公式来看，自动化测试的收益取决于 t 和 n，t 指的是节省下来的手工工作量，还是比较容易理解的。在字面上，n 是一份自动化测试案例重复运行的次数，那么在实践中，n 是什么呢？

聪明的你可能已经猜到了，n 是测试案例的稳定回归次数。软件的新功能开发出来后，第 1 次测试之后，第 2 次，第 3 次到第 n 次，都是对第 1 次的回归。它们都是重复的工作，应该被自动化替代。

你看，从 ROI 公式，我们很容易推导出业界熟知的经验**“自动化测试是用来做回归测试的”**。自动化是开发出来不会只运行一次，除非它的 t 特别大，实现了手工测试做不到的事情，比如单元测试、性能测试。

我们再回到回归测试，n 作为回归测试次数，对自动化测试工作有什么启发呢？它的作用很大，因为它能帮助我们量化地去回答一个 “做不做自动化测试” 的关键问题

一个测试案例 A 做不做自动化测试？首先要看看它的 n 能有多大。

假设软件发布周期持续一年，每两周迭代一次，每次迭代都需要一次测试，那么在这一年里，需要回归测试次数至少是 365/14=26 次。如果还考虑一些紧急 feature、patch 的发布，那实际的回归次数要大于 26 次。这样，我们就能得到一个 n 的估算值，比如说 30 次。

得到估算值后，你的决策不再依赖直觉，而是有了可量化的思考逻辑。30 次能不能收回来成本？能！那这个测试案例 A 就可以搞自动化。不能，你就面临亏本风险，自动化一顿操作猛如虎，测试工作还是苦，这是项目组里的每一个人都会感受到的。

刚才说的是基于软件发布周期不变的情况下，如何估算回归次数 n。在实际工作中，自动化测试一旦做起来，带来的变化是：测试执行时间变快，软件发布周期缩短，又反过来增加回归次数 n，自动化测试的收益也在增加。

这里我们又得到一个结论：软件发布周期变短是自动化测试 ROI 提升的产物。

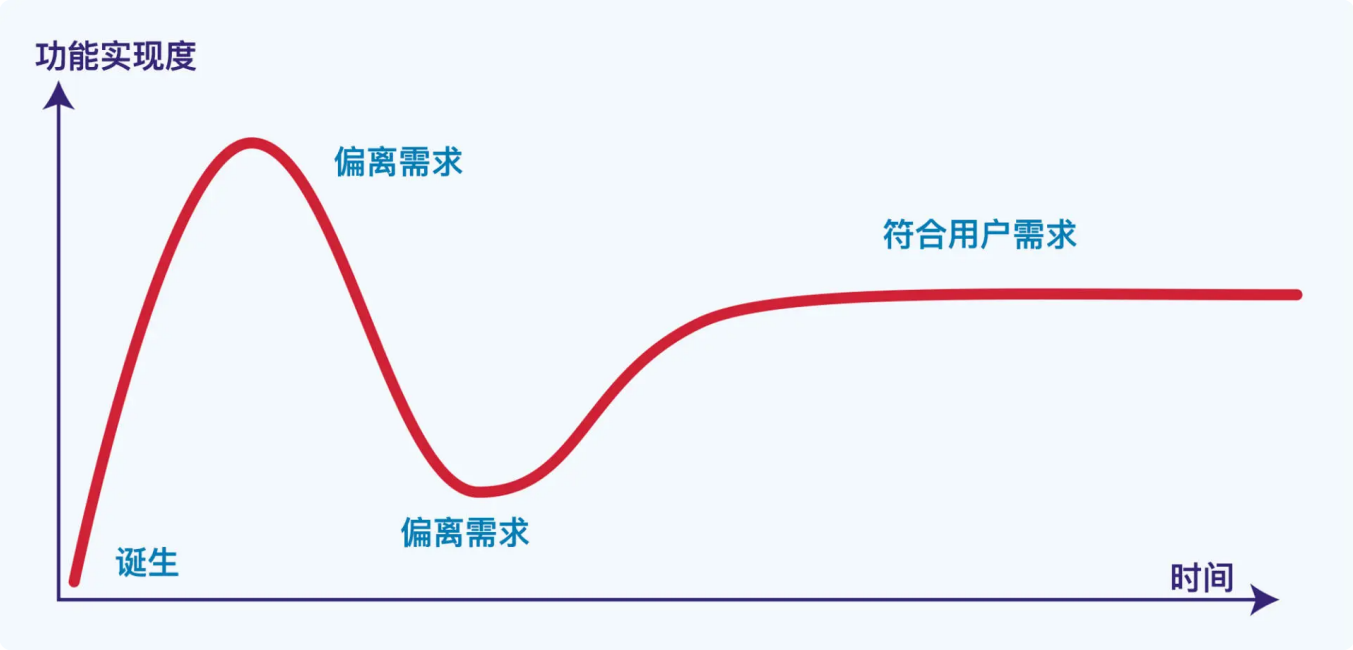
总结来说，只要我们把注意力关注放在 ROI 上，后面的好处都会相继而来，测试质量提高了，发布周期缩短了，团队也更加有信心了。

实践中，冒烟测试是你自动化的开始

紧接着，咱们再考虑下一个问题，测试案例 A 需要 30 次回归，是不是在刚引入新功能 A 的第 1 次迭代，就开始运行自动化？我的答案是要根据情况来判断。

上面说到 n 是测试案例的稳定回归次数，注意**稳定**这两个字，它代表功能 A 已经稳固下来，不再变了。更精确地说，功能 A 即使有变化，但是变化规律已经可以被自动化测试吸纳。这种情况下，自动化测试运行才能发挥效益。

这里你可以看看后面这张图，画的是加特纳的技术成熟曲线，它也可以用来描述软件功能的发展过程。



加特纳技术成熟度曲线

通过加特纳成熟度曲线可以看出，新功能在产生初期，一般是不稳定的，和它的预期有一个差距。经过几轮调整后，才会进入到一个平缓的阶段，这也是稳定回归测试的阶段。而不同类型的软件，它的功能成熟时间长短，变化剧烈程度可能是非常不一样。

有的软件是做标准化产品的，比如专业性强的 B 端财务软件，计税模块发布出来就很稳定，我们采取的策略是在第 1 个版本做计税模块的自动化。

有的互联网软件第 1 版是投放试验性的，我看过国外一个招聘网站经过产品设计，AB 测试多轮后，打磨了 x 版才稳定下来简历模块，那么这时的策略是在 x 版本进行简历模块自动化。

还有的生命周期比较短的软件项目，虽然有迭代，但功能一直无法稳定，那可能需要考虑完全手工测试，根本不需要自动化测试。其实这些都可以通过 ROI 模型讲得通。

到这里，咱们总结一下我们通过 ROI 得出的三个核心观点：

1. 自动化测试是用来做回归测试的。

2. 自动化测试从哪里开始？实施顺序从 ROI 高到低，也就是（给定一个软件系统），优先做回归次数最高的那部分功能，先做自动化回归次数最高的案例，再做低的，直到 ROI 等于 1 的案例。在功能模块的初期，可以考虑先做手工测试。

3. 自动化测试什么时候开始？功能模块稳定的时候。

实际上，有一个很好的测试实践可以匹配上面的要求，那就是冒烟测试。冒烟测试是测试用例的子集，用来验证系统中基础的、影响发布软件的功能。甄选冒烟测试的一个常用办法就是二八原则。

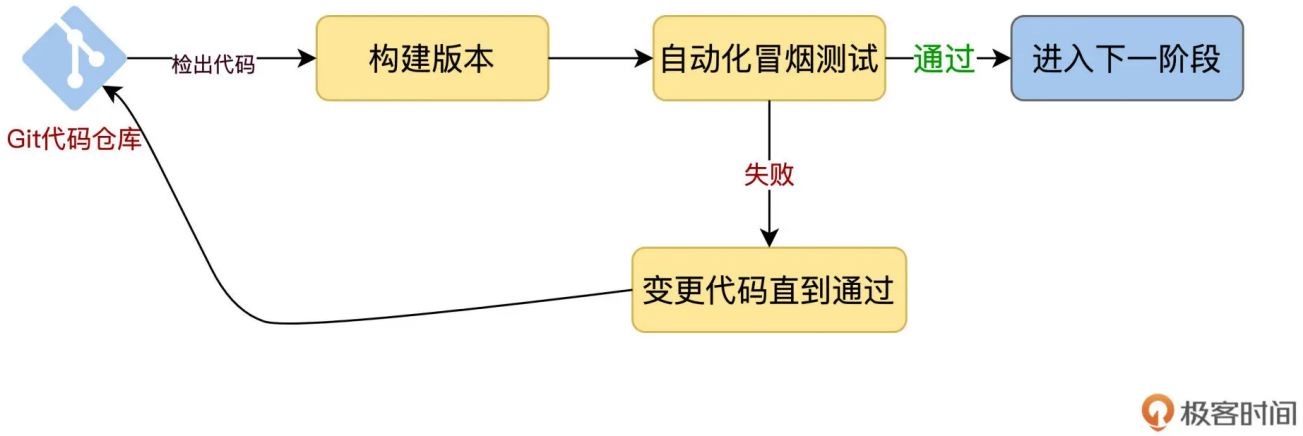
二八原则又叫帕累托原则，在因果关系中，仅有 20% 的因素会影响 80% 的结果。它在各个领域都有体现，比如在市场营销领域，80% 的利润是由 20% 的用户创造的；在经济学里，80% 的财富掌握在 20% 的人的手里。

在软件领域，80% 用户，常用的是系统中 20% 的功能。冒烟测试覆盖的这部分 20% 功能，是常用的，一般也是核心的，最先被开发出来的。所以，它同时满足稳定和回归次数高两个特点。

进而我们就可以得到推论：**在实践中，可以设定目标，冒烟测试 100% 自动化。**这时，自动化测试就可以和手工测试配合，形成一个新版本发布 + 冒烟测试的简单流水线。

如图所示，先从代码管理工具比如 CVS、Gitlab 中的开发分支中拉取代码，build 构建，做一轮冒烟测试。如果冒烟测试通过，开发分支可以 merge 到发布分支，如果冒烟测试失败，那开发人员必须修改代码，直到冒烟测试通过。

这个流水线 Pipeline 充分利用了自动化无人参与，执行速度快的特点，可以帮助开发人员在第一时间验证代码的正确。由于是每次分支归并都会调用冒烟测试，所以 n 的次数高，自动化测试的 ROI 也会高。



新版本发布+冒烟测试流程图解

小结

今天这一讲，我们通过一个投资产出视角来观察自动化测试，它的成本是什么，它的产出是什么，还学习了 ROI 的计算公式。

我们通过 ROI 的收益规律，不仅可以推导出自动化测试业界的已有共识，比如：“自动化是用来做回归测试”，“冒烟测试优先做自动化”……而且，我们还能挖掘出一些新的合理观点，比如：“ROI 从高到低，来做自动化测试”。

这说明业界的实践已经有意或无意地践行 ROI 规律，因此可以说，**ROI 是一个自动化测试项目的隐式命脉**。

同时，我们又详细介绍了 ROI 公式的因子 n，测试案例的回归次数。在实践中，找到 n 来估算 ROI，能帮你判断一个案例该不该做自动化。

ROI 公式里，除了 n 还有其它因子。在后面的课程中，我们再一一介绍其它因子，像 m 维护成本，现在这个概念看起来还有点模糊，我还会帮你把它分解，直到可操作和可度量的粒度，让 ROI 的方法论更有效地指导你的工作。

思考题

学习 ROI 之后，你可以从开篇的三个问题里选择一个或多个，试着回答一下。

“产品要上线了，QA 人手紧，能不能搞一下测试自动化，减少点人手?”

“什么？你还要再增加 2 个自动化测试开发工程师来完成这个项目，怎么算出来的？”

“听说你开发了个什么自动化脚本，它给公司带来了什么价值？用量化的数据给我讲一讲。”

欢迎你在留言区记录你的收获和思考。如果这一讲对你有启发，也可以推荐给身边更多同事、朋友，跟他一起学习进步。

分享给需要的人，Ta订阅超级会员，你最高得 50 元

Ta单独购买本课程，你将得 20 元

生成海报并分享

赞 4 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 开篇词｜做性价比最高的自动化测试

下一篇 02｜3KU法则：如何找出最优自动化实施截面？

精选留言 (8)

写留言

小火车
2022-03-21

ROI 是一个自动化测试项目的隐式命脉，说的太对了，可以说现在所有的工作，包括计算机行业外的工作，都完全离不开ROI

作者回复: 是的，本专栏是想帮助同学们在工作实践中“做最好性价比的自动化测试”，把我们有限的精力，投入到ROI最高的地方。不要做“重复的”，“无效的”自动化测试。

4

追风筝的人
2022-03-22

note 1. 自动化测试是用来做回归测试的 2.软件发布周期变短是自动化测试 ROI 提升的产物。 3.ROI 是一个自动化测试项目的隐式命脉。

展开

作者回复: 非常到位！

1

郑小菜

2022-03-26

有个问题请教下，文中提到的“冒烟测试”，我是否可以理解为产品主流程？还是说指的就是持续集成中某个要提测feature的核心case？

作者回复: 我试着get一下你要说的意思哈，如果产品是基于work flow的，那么冒烟测试应该是产品中那部分最常用最基本的flow path，应该就是你说的“产品主流程”。所以，冒烟测试的用途是这样的，如果它失败了，其他的功能都不需要测了，直接返回开发修改。冒烟测试也可以用在很多场景，提测feature是场景之一，后面1.6讲，也会提到线上监控使用冒烟测试。

willwchen

2022-03-23

自动化测试提升节约了多少测试时间的度量标准，感觉只适用于测试团队内部度量，怎么把这种价值传递到整个业务团队，能让业务团队感知到自动化对业务带来的价值，这个是目前最大的挑战，现实是业务团队对提升的测试效率感知不强，不感兴趣，2个100万的自动化工程师，对业务团队的感知提升，可能不如4个50万的业务测试工程师，对比衡量，是不是4个50万业务测试工程师价值更大，这4个业务测试工程师除了本身测试工作，还...

展开

作者回复: 看出来可能你负责的是领域性软件，业务性比较强。测试人员在团队种被赋予了业务专家的属性，这时，能引起团队共鸣的一个自动化测试价值，是把业务知识规则化表达出来，先让知识可表达，可存储。我在1.5里会说到这个！

IT蜗壳-Tango

2022-03-23

ROI的维护成本的确很难确定，除了维护成本还有作业报告的分析等因素

太匆匆

2022-03-22

冒烟测试进行自动化确实是最合理的选择，现在有了ROI这个武器，拒绝自动化测试无限扩大就有底气了。天下苦无限自动化测试久矣！

作者回复: 说得好！

追风筝的人

2022-03-22

1.快上线了 应该不能吧 来不及了 下次 2. unknown 3.ROI, 降低了重复执行用例的时间，提高了效率。 为后期自动化回归测试用例，当n> flag 适合做自动化， ROI>1，为盈利状态；

展开

woJA1wCgAAUtR8JWp4a3...

2022-03-22

开发没有完成的情况下，怎么用自动化去写冒烟用例给开发自测呢

作者回复: 这个可以有，需要和开发一起协作
单元测试领域的TDD测试驱动开发
集成测试领域里的基于契约的测试
UI测试领域可以和开发复用组件库

共 2 条评论 >